Project 5
Team 10
Section 12:30pm-1:45pm

Tomas Kvapil, tkvapi2@uic.edu
Damian Ortizflores, dortiz36@uic.edu
Raja Patel, hpate85@uic.edu J
avier Rojas, jrojas39@uic.edu

Section 1 – The purpose of our project

This project is an original superhero card game in which a player is given 5 cards of superheroes/villains to battle against another player. Each 3 battles won wins the game. The problem we saw was that there was a lack of superhero games card games that pitch both DC, Marvel and much more brands together. We only hope to have people play our game and experience it first hand.

Section 2 – The high level entities

Our project consists of two folders two run two separate programs. One folder for the server and one folder for the client/player. For a game to run, there needs to be one server running and two or more players. There needs to be at least two players to start a game because they battle against each other. The server folder contains: Card.java, Dealer.java, Game.java, MainGui.java, NetworkConnection.java, Player.java, and Server.java. The client folder contains: images folder, Client.java, MainGui.java, and NetworkConnection.java. The client folder is very simple because all the clients do is send information to the server where or logic is handled. Main is found in MainGui where all the GUI components are found as well. This is where all the images are used in the image folder. The image folder just contains all images for every superhero in our game. The NetworkConnection and Client classes are used to setup connection to a server and to communicate back and forth the the server. The information is then displayed through the GUI. The server folder is a lot more complex because this is where everything is handled from all the clients. For our game to function properly we need a card class to create cards that have all the info on them. Then a dealer class that creates a deck and has all the deck functionalities. Then a player class to be created for each new client that connects. Then a game class that keeps track of the whole game. The MainGui is where main is and where information is handled with gui and clients messages. NetworkConnection and and Server are setup to create a server socket and listen and send messages. For the bulk of things, the server is do mainly all the work.

Section 3 – Low level design

Client side

NetworkConnection and Client:

These two classes work together to formulate a proper client program that connects to our given server. The port and IP address are already embedded in the code so the client connects to our server automatically. This is also where the client listens for input from the server and sends output to the server. They communicate the information back to the GUI where the it makes the correct updates.

MainGui:

This class is the class that makes the client look so good. This is where all the GUI components are handled and made visible to the client. Depending on what information the client receives, the GUI will make the adjusted changes. In totality this class only displays the information from the server. And the server sends information to specific clients so that not all the clients look the same.

Server side

Card:

This class contains all the information for a single card. It also contains the enum for the superhero names to be created. This card is very crucial because it contains the attack, defense, special, and name of the card. This class has the most instances because when a deck is created it creates all the cards for all the superheroes

Dealer:

This class contains all the functionality for the deck. It has the ability to create a new deck which contains all the superhero cards. It then also has the ability to shuffle the deck so that no two decks are the same. Finally it can scratch a current deck and create a whole new one for when a new game is requested

Player:

This class is created automatically when a new client joins. When a player joins it will assign them with an id that then is contained in the player class. This is so that we can get the id of the player and send to that id through the server. The player also has their hand. This is so that the server has the information on what each player has in their hand. Finally it keeps track for how many points that they have scored.

Game:

This is where the game logic is handled as well as keeps track of the players in the game. This happens when a player joins then they are add. This is also where a random challenge is made and keeps track of who is the challenger and who is the contender. Then it also contains the game logic of a battle. So once the challenger and contender have chosen their cards, it will compare the two and return who has won the battle. Finally it has the capability to reset the game when it is needed.

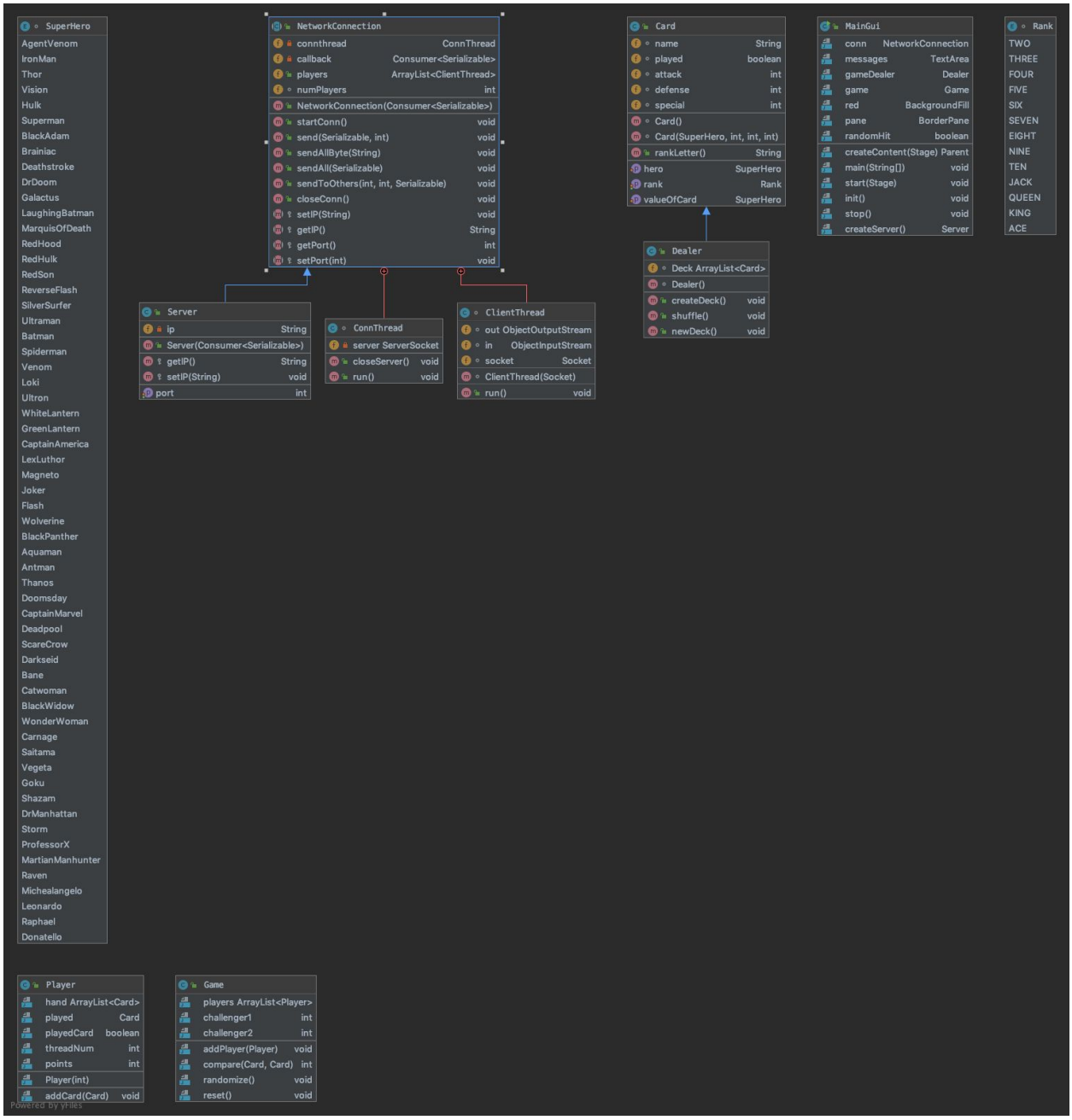
Server/NetworkConnection:

These two classes work together to formulate a proper server program that sets up a server to start listening for clients. The port is already embedded in the code so that no extra steps are needed when the program is run. This is also where the server listens for input from the clients and sends output back.

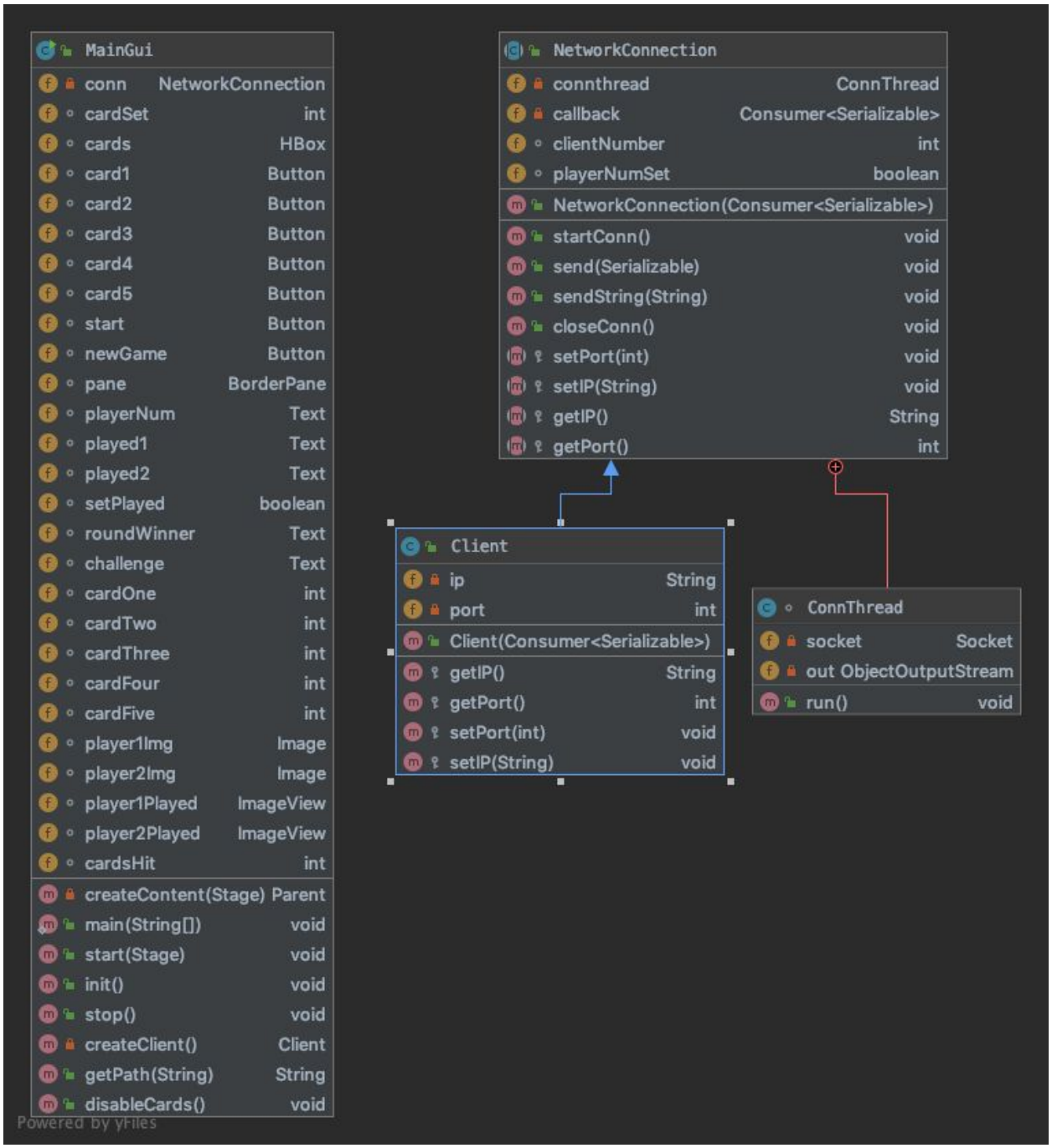
MainGui:
The main purpose of this class is for hosting main as well as connecting all the information. There is a GUI for the server but this was mainly used for error handling. This was done using the text area to see what the server was sending and receiving

Model

Server UML Diagram

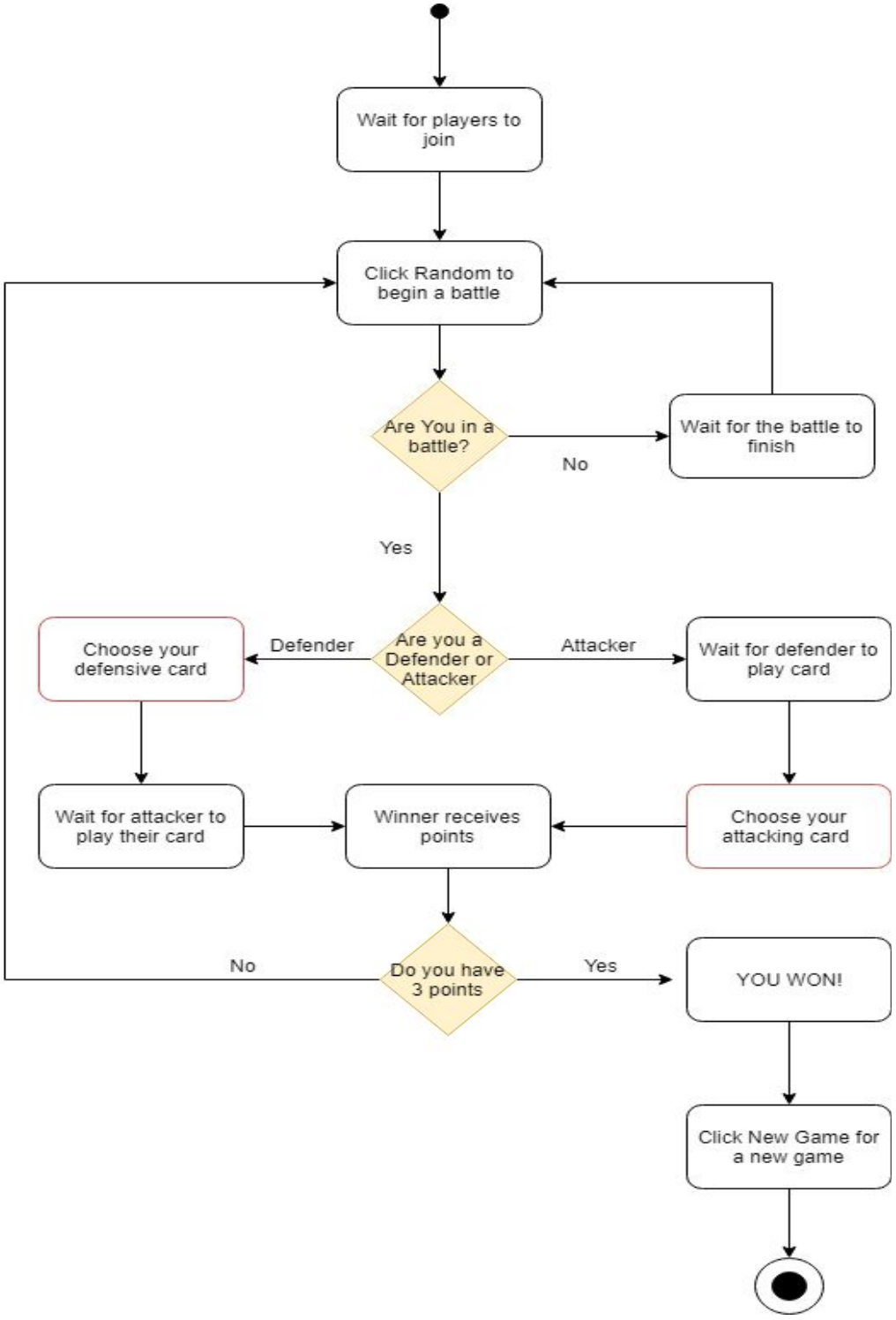


Client UML Diagram



Interactions

Activity Diagram



Section 4 – Benefits, assumptions, risks/issues

The overall design and implementation of this project is pretty well done. It works excellently when players join and then playing against all the players in the game. The only flaw of the game is that once a game is created, the players cannot leave or else it will cut the code. However, other than that it was perfectly fine. It works fine and it looks super good too so that the client have a visually appealing time playing the game. This game was made in JavaFX so it was made for people to know who they are playing against have multiple players just login to our server and start playing our original game.

Conclusion

This project consisted of many moving parts from the client, to the server, from implementing the GUI and having them all work together to make this game possible. Not only was this game challenging but it was also rewarding. Seeing it work and having the opportunity to play something created from the bottom up was the most rewarding and memorable.